AD-A043 267 HUMAN ENGINEERING LAB ABERDEEN PROVING GROUND MD F/G 5/9
COMPUTER-GENERATED DISPLAYS ADDED TO HEL HELICOPTER OPERATIONAL--ETC(U)
MAY 77 G L HERALD
UNCLASSIFIED HEL-TM-18-77 NL

| OF |
AD
A043 267

END
DATE
FILMED
9-77
DDC

12

Technical Memorandum 18-77

COMPUTER-GENERATED DISPLAYS ADDED TO HEL HELICOPTER OPERATIONAL TRAINER

Gordon L. Herald

May 1977
AMCMS Code 672716.H700011

U. S. ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
Technical Memorandum 18-77

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
(6) COMPUTER-GENERATED DISPLAYS ADDED TO HEL HELICOPTER OPERATIONAL TRAINER.

5. TYPE OF REPORT & PERIOD COVERED
(9) Final rept.

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
(10) Gordon L. Herald

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
U.S. Army Human Engineering Laboratory
Aberdeen Proving Ground, MD 21005

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
AMCMS Code 672716.H700011

11. CONTROLLING OFFICE NAME AND ADDRESS
(14) HEL-TM-18-77

12. REPORT DATE
(11) May 1977

13. NUMBER OF PAGES
39 (12) 42p.

14. MONITORING AGENCY NAME & ADDRESS (*if different from Controlling Office*)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

172 850

18. SUPPLEMENTARY NOTES

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Displays
Imaging
Simulation
Helicopter Simulators
Computer Generated Imagery
Aircraft Panel-Mounted Displays

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

A Singer-Link Helicopter Operational Trainer has been interfaced with the U.S. Army Human Engineering Laboratory's Command Control Simulator computer to provide the helicopter trainer with real-time computer generated cathode-ray-tube imaging displays. This report discusses the equipment configuration, data collection, displays and the hardware and software problems associated with this development.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

AMCMS Code 672716.H700011 Technical Memorandum 18-77

COMPUTER-GENERATED DISPLAYS ADDED TO HEL HELICOPTER OPERATIONAL TRAINER

Gordon L. Herald

May 1977

APPROVED: [signature]
JOHN D. WEISZ
Director
U. S. Army Human Engineering Laboratory

U. S. ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland 21005

ACKNOWLEDGEMENTS

Gratitude and recognition are due to Mr. Richard S. Camden and Mr. William B. DeBellis for their expert assistance in many facets of this system integration development.

CONTENTS

TABLES

COMPUTER-GENERATED DISPLAYS ADDED TO HEL HELICOPTER OPERATIONAL TRAINER

INTRODUCTION

In late 1975, the US Army Human Engineering Laboratory (HEL) initiated a two-phase effort to interface its Helicopter Operational Trainer (HOT) to the Command Control Simulator, thus extending the human engineering aviation design support capabilities to provide:

1. Real-time, dynamic, computer-generated visual displays.

2. A means of measuring flight crew performance as it relates to new and proposed cockpit displays.

3. Automated data collection.

4. Replay capabilities for analysis and crew debriefing.

The extension of the imaging and computation capabilities of the Command Control Simulator (CCS) System will permit effective studies to proceed with regard to alternate sets of visual cues to be presented on head-down panel-mounted displays or helmet-mounted displays. The system is sufficiently flexible so that head-up, out-the-window displays could be provided with the addition of a projection cathode ray tube (CRT).

The development of this capability has required a significant expenditure of effort in programming and systems integration to enable a limited computer configuration to provide the support functions enumerated above. Exploratory software has been developed to provide proof of the real-time dynamic imaging capabilities for both line-drawn and solid-surface computer generated images.

Figure 1 shows the cockpit view of a line-drawn dynamic flight display presented on the panel-mounted CRT.

The following discussion will describe the major system components and some hardware and software problems encountered in this development.

Equipment Configuration

The HEL Command Control Simulator (CCS) system evolved slowly from its initial 1969 configuration consisting of a Varian 620i computer with three monochrome CRT's, paper tape and teletype input/output and assembly language programming. The present system, shown in block diagram form in Figure 2, is composed of three categories of equipment—digital, analog and imaging. FORTRAN programming methods and a computer operating system have replaced the original assembly language methods and have reduced program development time.

Figure 1. Picture of HOT with dynamic flight display on CRT.

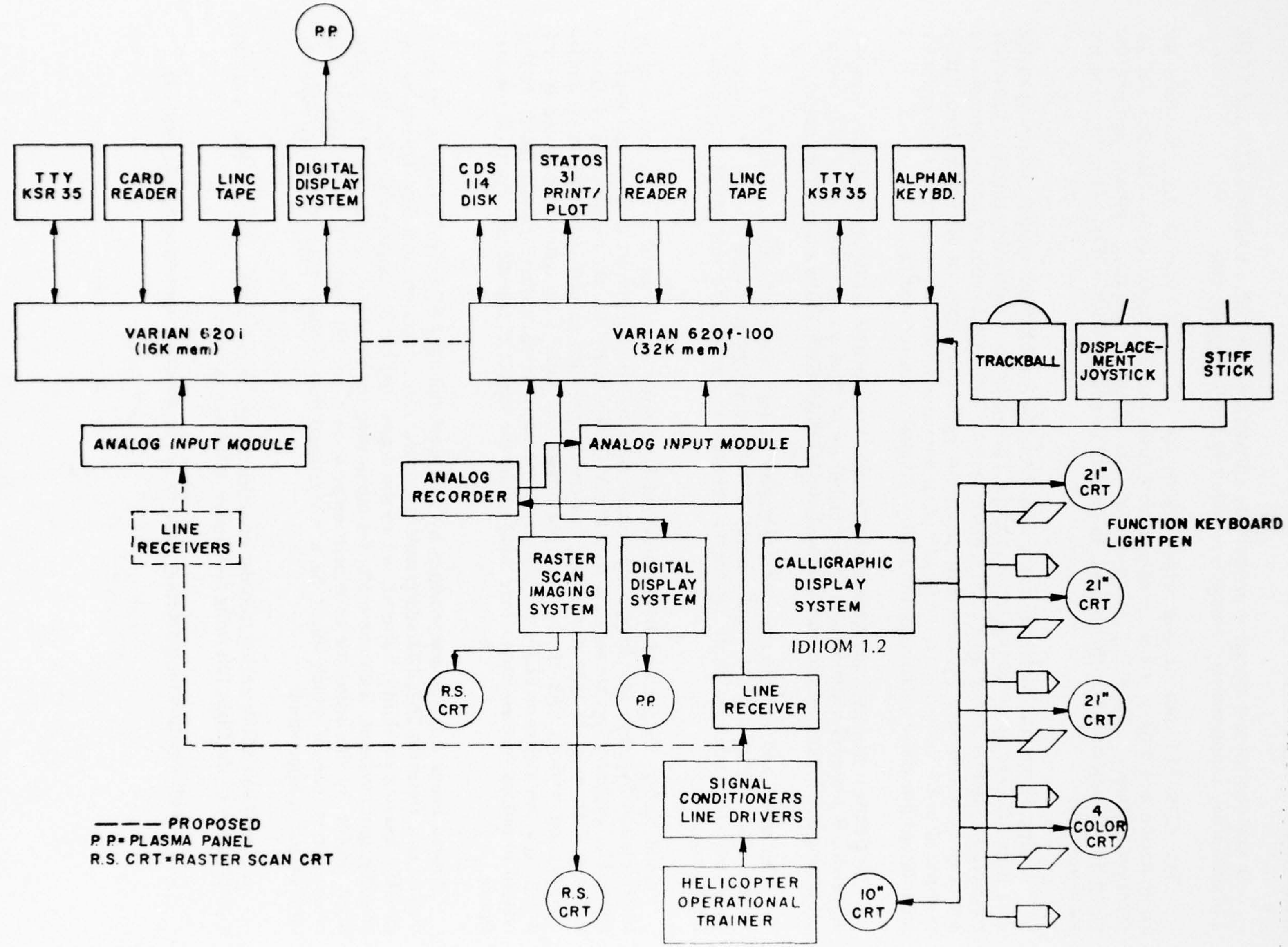

Figure 2. System block diagram.

Major System Components

Digital control and storage is provided by a VARIAN 620f-100, a VARIAN 620i and a CDS 114 disk drive. Supplementary storage is provided by LINC magnetic tape.

The CDS 114 disk drive provides storage for 29 million bytes of data. Routines for computation and display as well as imaging data bases and test results reside in the disk storage. Dynamic memory allocation, provided with the software operating system, permits the computer, in association with the disk, to execute programs which will not fit in the available main memory.

Analog devices are interfaced to the CCS through an analog input module. Incoming analog data is multiplexed at the analog input module. The multiplexer currently provides a capability to select 16 external analog signals and is expandable to 256. Signal sampling rate is controlled by preset clock or computer program. After analog-to-digital conversion has been completed, data is transferred to the computer on the input/output lines as permitted by programmed sensing or by an interrupt procedure initiated by the analog-to-digital converter module.

Line-drawn displays are generated through the IDIIOM calligraphic display system. Appendix B provides a summary of the capabilities of this system. The IDIIOM 1.2L display processing unit bidirectionally interfaces with the VARIAN 620f-100 as a display file buffer.

The speed and precision of the calligraphic system make it highly desirable for human factors studies that do not require solid surface computer-generated images; i.e., dynamic character and symbology study related to information transfer and subsequent man-machine interaction.

Solid surface image capability is provided by a raster scan imaging system developed by Interpretation Systems, Inc. Images are presented on television monitors in black and white or color. The image buffers are separate from the VARIAN 620f-100 core memory. This system is different than most in that dual image buffers are provided to permit massive image buffer modification without loss of the displayed image. The raster-scan system is oriented toward providing a simulation of LLTV or FLIR nap-of-the-earth terrain displays. Human factors testing related to display in these technology areas will be the subject of considerable efforts in the near future.

Stored image displays are provided by an Owens-Illinois 8.5" by 8.5" Digivue flat-panel display unit featuring 262, 144 addressable points and rear projection capability. This device has dynamic imaging capability; however, not to the degree that can be attained with the raster-scan or calligraphic systems. It can be useful for monitoring the simulator ground track and vertical flight profile by the instructor or experimenter or as a cockpit display for a variety of data presentations, one of which might be a navigational map display. Digivue specifications are summarized in Appendix B.

A Honeywell 5600c analog recorder provides storage for analog data from the HOT and also enables replay of the flight following conclusion of the test mission.

Detailed information on the display systems can be obtained from references (1) and (2).

The final large element of the CCS system to be considered, the Helicopter Operational Trainer, is the most recent addition to the system and one which represents, to a high degree, the greatest potential for study of man-machine interaction and the resulting human factors aspects.

Complex integrated aircraft displays are being developed by various Army laboratories for which little or no human factors data exist, especially for those aircrew tasks related to NOE flights and integrated avionics panel displays.

The Helicopter Operational Trainer (HOT), when attached to the CCS provides an inexpensive means of obtaining human factors data related to cockpit displays.

The HOT (Figure 1) includes a cockpit, a hydraulically actuated motion system with two degrees of freedom and a computer. The interior of the cockpit provides the controls and instrumentation representative of a single-rotor helicopter equipped for IFR procedures. Engine sound effects add to the cockpit realism.

Panels and controls are modified by the HEL as required to support human engineering design tests and research.

HOT characteristics are provided in Appendix A.

HOT DATA CONVERSION AND STORAGE METHODS

HOT Analog Data Conversion and Sampling

The HOT analog data conversion was simplified by making all HOT outputs conform to DC levels through hardware signal conversion/conditioning at the line driver modules. The analog line receiver outputs are connected by coaxial cable to a multiplexed analog to digital converter module. Data samples may be made randomly or sequentially as commanded by computer program control. The selection of random or sequential mode is determined by the programmer and is influenced by the rate and type of data to be collected. The time between samples also can be selected by the programmer. Programs which require long computation periods after each data sampling pass will generally require a data sample on program demand. The computer generated flight display program (4) operates in this mode due to the ½-second computation delay required to perform matrix multiplication, synthesis, line clipping, projection and display after each data collection pass. Programs which have sufficient idle time after each data collection and computation pass may be moved from their idle state by the interrupt capability of the analog-to-digital converter module at, for example, every 100 ms. Interrupt frequency is selected by the programmer but cannot, however, be of shorter duration than required by the computation process which follows.

The end result of the data-conversion process results in the execution of two computation procedures. The first procedure will systematically and chronologically process data related to flight-crew performance and store the results on the disk for later statistical analysis. The second procedure results in modifying a display file which permits the displayed image to be updated.

Test Mission Data Storage and Hardcopy Output

Data storage for test missions is provided at two levels. The first level is analog data from the HOT and the second level is digital data pertaining to test-mission activity, and activity assessment.

A Honeywell 5600c records a number of selected analog signals from the HOT to provide replay capability. At 3-3/4 ips, a single reel will provide storage for 2 hours of testing. Slowly varying DC level analog signals from the HOT require the use of FM record and reproduce circuits.

Analog signals may also be digitized and stored on LINC tape and disc to provide rapid access to replay selected segments of the test flight.

Selected analog channels are digitized by the analog-to-digital system and transferred to the digital system. A portion of this data is used by special software routines to generate data for display files. Other software routines assess aircrew performance, simulate radar-warning alerts, ground-proximity alerts, aircraft emergencies, etc. This data moves from core to disk as required by the routines currently executing in the digital systems. Long-term digital storage is provided by the supplementary LINC tape system.

Hardcopy Output

Hardcopy output on the CCS system is provided by a Statos 31 printer/plotter. During program preparation or during data analysis runs it serves as a line printer. During test exercises or replay, it can provide hardcopy of the CRT display.

HOT PANEL MOUNTED CATHODE RAY TUBE DISPLAYS

A high resolution 10-inch diagonal monochrome cathode-ray-tube display is panel mounted in the HOT. The CRT is connected by an 8-foot flexible cable to the electronics unit mounted externally to the HOT (Figure 3). Some of the pertinent specifications for the panel mounted line drawing display are provided in Appendix B. Specifications for the raster-scan panel mounted CRT are incomplete at this time.

Three large screen (21") calligraphic monitor stations with interactive devices for experimenter control, monitoring and program development are located in the CCS room (Figure 4). Development efforts requiring color may utilize the four-color penetration tube CRT.

PROGRAMMING CONSIDERATIONS

What appears at first to be an overwhelming array of hardware is easily brought under control by computer programs. Special systems and applications programs written at HEL perform within the HIGHER disc-based batch-operating system developed by Information Displays, Inc. The operating system permits the generation and control of interactive display programs to be written in FORTRAN and also permits imbedded assembly language statements

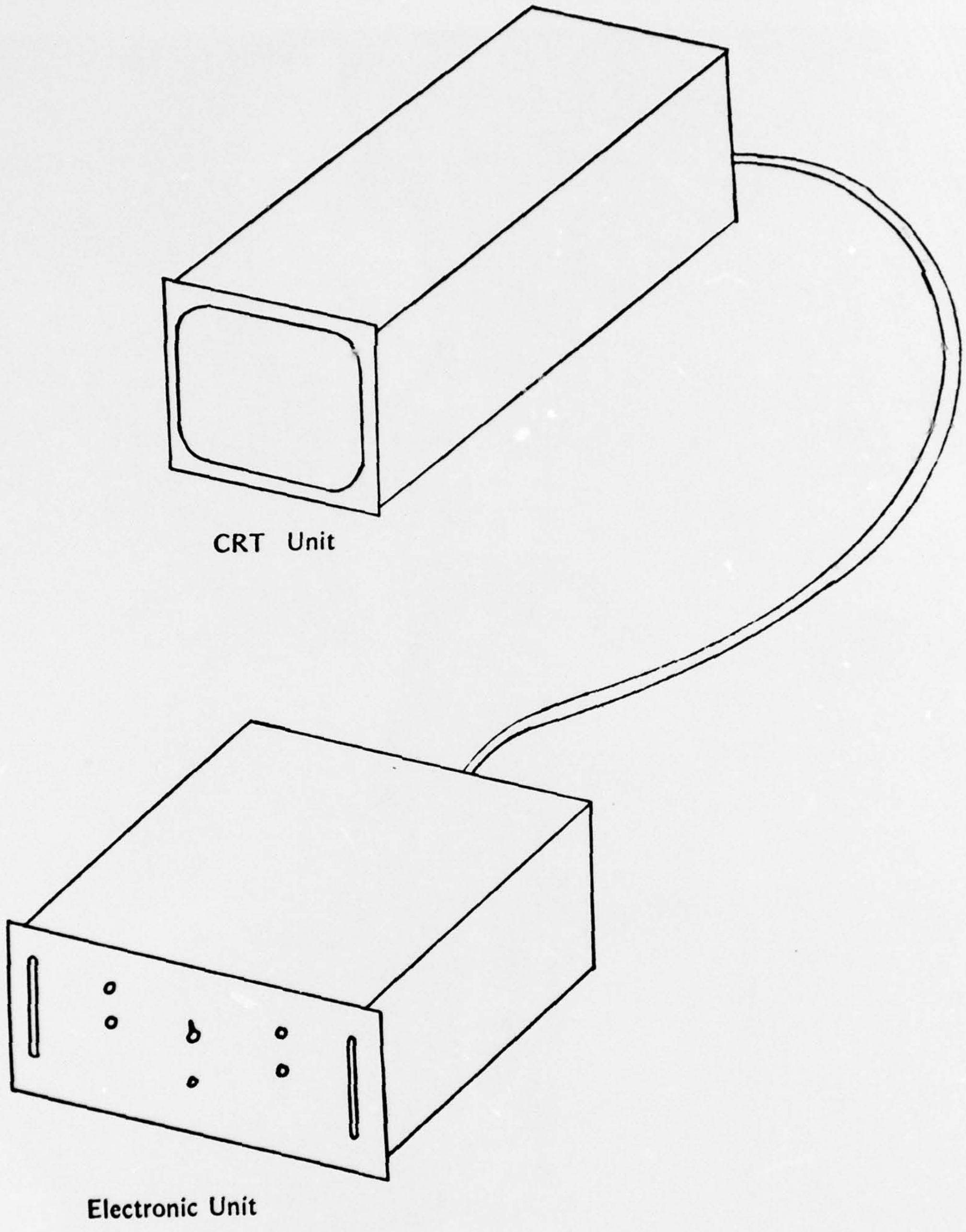

Figure 3. Outline drawing, Multifunction Display Indicator.

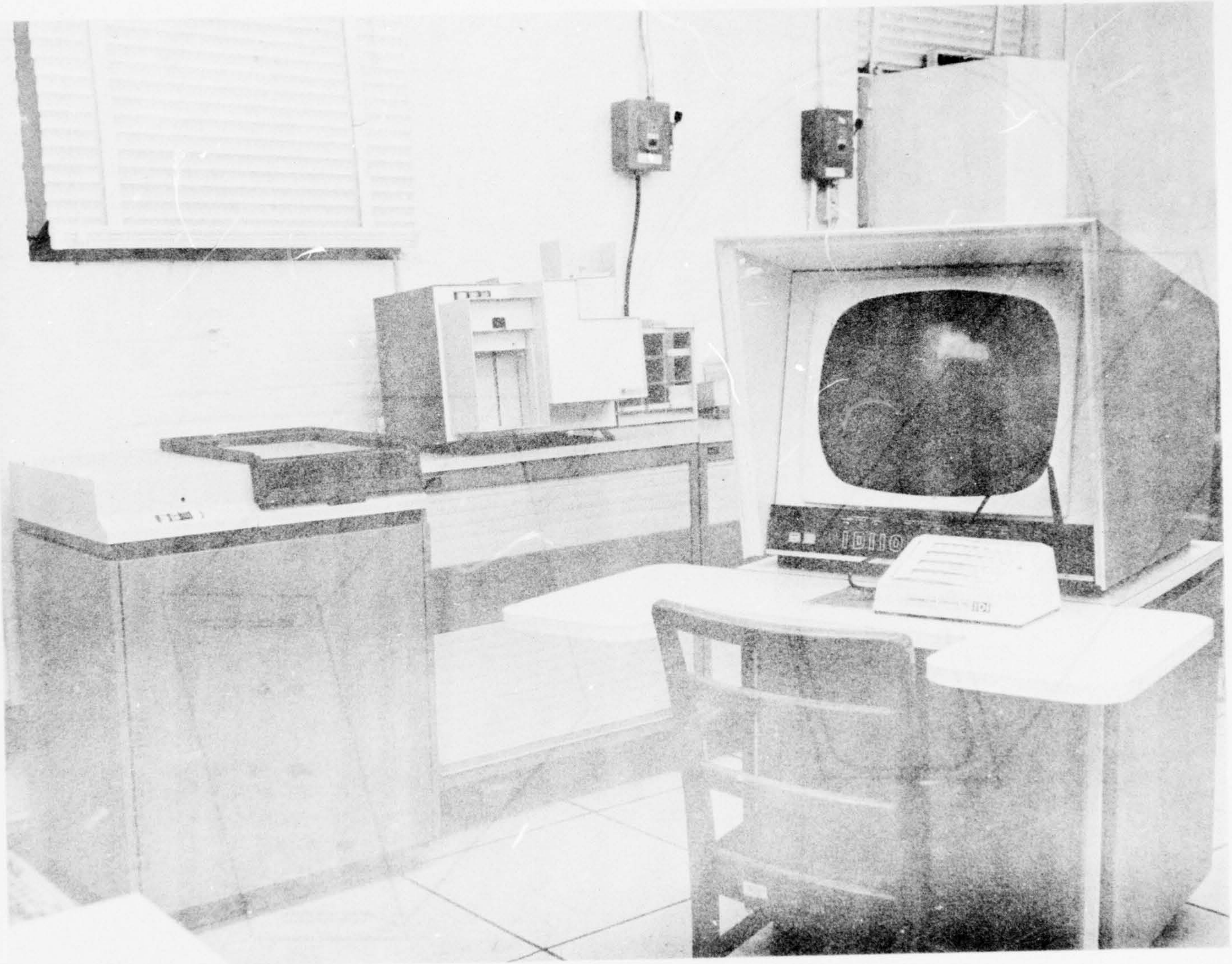

Figure 4. Twenty-one-inch view of monochrome console in CCS room.

in the coding whenever an operating improvement justifies its use. Special IDIIOM FORTRAN Graphics subroutines permit the development of interactive display files using FORTRAN. Routines, developed by HEL, enable analog-to-digital conversion processes to run under control of a real-time clock.

Programs developed by HEL permit physiological data to be collected during experimentation on analog tape and then presented for preview on the CRT's in an interactive mode for analysis prior to processing by math routines. CRT presentations of relevant data may be plotted. Reference (1) provides details regarding this type of application.

Software for Phase II applications requiring the raster-scan hardware for solid-surface displays is being considered for specification and development at this time.

REAL TIME SYSTEM RESPONSE HARDWARE AND SOFTWARE CONSIDERATIONS

One of the most complex problems encountered in integrating visual devices with complex motion-based simulators is the realization of real-time response.

Within the sphere of simulator applications, at HEL we consider a realtime response as one which meets the following definition:

> "A response is considered real-time when the computations proceed at such a rate that the process underway is influenced without creating a perceptable delay to an operator in a man-machine loop."

For practical and economic reasons, it is not always possible to achieve the ideal expressed by the foregoing definition. However, with regard to training and human factors testing, the apparent delay perceived by an operator in a man-machine loop must be sufficiently small so as not to cause the operator to perform differently than if the delay were non-existant.

Sources of Response Lag

The HEL helicopter simulator will have three sources of motion cues for the pilot. The standard panel instruments represent one set of cues. Physical motion cues in the form of pitch and roll are created through a hydraulic motion system. Visual cues are presented in the form of a panel-mounted CRT.

The physical motion and instrument cues are adjusted and correlated to produce the desired response for the aircraft being simulated. The motion computer, block 2 in Figure 5, has the task of generating the necessary electronic signals for the panel instruments and hydraulic system. The lag time from pilot input to instrument or physical motion output is on the order of 100-400 milliseconds. These are desired values and are built into the system to characterize the actual aircraft being simulated.

Visual simulations, especially those requiring psuedo 3-D imaging, expend the major portion of the computer-processing cycle, in matrix multiplication operations on the data base. Special hardware array processors are available which can permit high speed processing of the imaging matrices. Their cost, at $20,000 and up, begins to approach the cost of the main frame of many small computers. Good 3-D imaging results can still be obtained, however, without the specialized hardware.

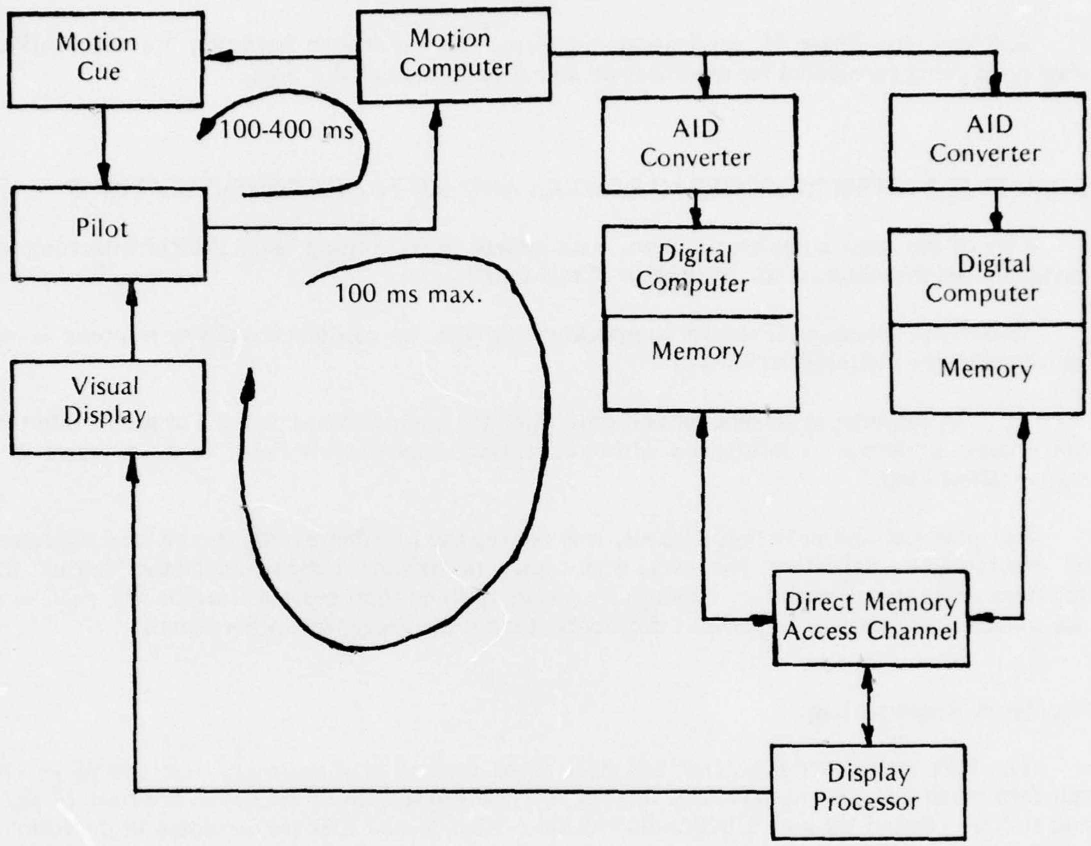

Figure 5. System loop delays.

Software Methods to Compensate for Hardware Limitations

To avoid the use of special matrix hardware in small computer systems, a few software concessions can be utilized to decrease the processing time. Those available are integer arithmetic procedures, assembly language procedures, interlaced data-interpolation procedures, extrapolation procedures, parallel processing, and compiler efficiency improvements.

Processing routines which can be converted from floating point to integer, or fixed-point, can achieve speedups on the order of 20 to 30 times when implemented on minicomputers not having floating point hardware. Tables 1 and 2 provide some insight into the amount of speed improvement that fixed-point arithmetic procedures can provide. For example, the software floating-point multiply on the CCS requires 258 us. The same operation using fixed-point hardware requires only 7.86 us—a speed improvement of 34 times.

TABLE 1

Floating Point Arithmetic Processing Time[a]

Floating Point Add	212 us
Floating Point Subtract	212 us
Floating Point Multiply	258 us
Floating Point Divide	359 us
Floating Point Square Root	1400 us
Fixed Point Square Root	2230 us

[a]These procedure execution times were obtained using *arithmetic software provided* by the HIGHER operating system performing on the VARIAN 620f-100 computer. Times include memory store of result.

TABLE 2

Fixed Point Arithmetic Processing Time[a]

Fixed Point Add	3.00 us
Fixed Point Subtract	3.00 us
Fixed Point Multiply	7.86 us
Fixed Point Divide	7.86 us

[a]Execution times measured on the VARIAN 620f-100 with hardware multiply/divide. Time includes fetch and store of upper 16 bits of result.

The actual overall speed improvement is not usually as great as indicated by number comparisons derived from the tables. To permit operation over a reasonably large range of numbers, scaling procedures must be employed. This will require at least one more fixed-point arithmetic operation, thereby reducing the speedup, on the average, to about 20 times.

The sophisticated arithmetic hardware available in modern medium and large scale computers has relaxed the awareness of the problems which may be encountered in using fixed-point arithmetic procedures. Many digital computing books gloss over or completely ignore the topic. It is an important procedure, especially when minicomputers are applied to real time systems.

Several problems arise in the use of fixed-point procedures. The first problem, scaling considerations, is one of adding two negative numbers or subtracting oppositely signed numbers. These operations can result in numbers which exceed the word size of the computer. Division, without appropriate scale factors applied, may also result in a quotient which overflows. Another problem related to fixed-point operations in minicomputers concerns the word length. Most minicomputers use a 16-bit word length which may not allow sufficient precision to eliminate, completely, all floating-point calculations. The limited word length tends to increase truncation errors and becomes noticable in display applications by the motion quality.

Differences in procedure execution times can be significant depending on the efficiency and type of compiler used to convert the source program to machine language. The source language level used for programming also becomes a factor in the efficiency of the machine language program. Programs that are written in assembly language can be as efficient as the skill and determination of the programmer will permit. However, the complexity of most imaging programs requires, for practical purposes, that high level language such as FORTRAN and ALGOL be used.

The software system under which a program is compiled will also affect the execution time. Programs compiled under an operating system which provides dynamic memory allocation will not execute as fast as the same program compiled under an efficient standalone compiler. The lower execution speed results from additional coding required to support the memory management techniques of the operating system.

Execution time for four programs was measured at the HEL using the operating system FORTRAN compiler and again for a standalone FORTRAN compiler. Table 3 summarizes the results of the time comparisons. The program used for the execution time measurements are provided in Appendix G.

We would expect to find the operating system to be slower in every case because of the dynamic memory allocation capability, however, for peculiar reasons, this was not the case for routines one and four.

With regard to routine one, it was found that the supplier of the standalone system had made some expedient and crippling patches in the compiler to allow the use of hardware multiply/divide which resulted in excessive and unnecessary coding. This problem is also reflected in the execution time of routine four. The execution speed of the operating systems for routine two (floating point) could be improved with an improved floating point software routine.

TABLE 3

Compiler Code Execution Time Comparisons

Program	Compiler O.S.[a] (sec)	S.A.[b] (sec)	Remarks
1. Integer Routine	3.2	104.0	O.S. 69 percent faster
2. Floating Point Routine	77.5	60.5	S.A. 22 percent faster
3. Mixed Int/Floating Pt. Routine	80.5	70.2	S.A. 13 percent faster
4. Mixed Int/Floating Pt. Routine	85.2	89.2	O.S. 3.4 percent faster

[a]O.S. - Operating System

[b]S.A. - Standalone System

With due consideration to all improvements in both compiler systems, we can expect that the standalone system (S.A.) will always provide code which will execute 15-20 percent faster than that provided by the operating system. We arrive at these results from an estimated 5 percent improvement in the operating system (O.S.) floating point execution time and improving the S.A. integer execution time to at least equal the O.S. integer execution time.

For a given installation, there is nothing that can be done to greatly modify the existing central processing unit architecture. However, if an opportunity exists that would enable a choice to be made in the selection of a CPU for real-time imaging routines, there are architectural designs which will influence execution time and also directly influence compiler code efficiency. Word size, number of general purpose registers, number of registers available for arithmetic operations, floating point hardware, microprogramming capability, etc., will affect the amount of code generated.

Some small processors offer 24- and 32-bit word size versus 16-bit word size for others; seven accessible general purpose registers versus three accessible registers; hardware floating point add/subtract/multiply/divide square root versus hardware fixed-point add/subtract/multiply/divide.

Every additional general-purpose register that is provided by the central processing unit architecture can improve procedure execution time in several ways. The amount of register-to-memory swapping can be reduced. Arithmetic operations can be optimized by eliminating redundant load and store operations through retention of arithmetic results in registers. In general, compiler optimization goals that leave values, indexes, and address constants in registers for future use are desirable.

Word size has a significant effect upon code optimization and procedure execution speed. Larger word sizes will reduce the number of double-word instructions thus reducing the amount of code generated and directly reducing the number of memory accesses required. Larger word size also provides a versatile instruction set which can, in a single word, specify complex operations utilizing multiple registers.

A hardware floating-point arithmetic unit will probably do more to improve the execution time of real-time simulation programs than any other available computer option. The time to produce a floating-point arithmetic result can be reduced by a factor of 10 compared to floating-point software methods. A floating-point arithmetic unit will also permit a reduction in coding and some units will also permit some concurrent processing with the CPU.

Real-time programs can benefit by the use of optimizing compilers. This is a compiler feature which rearranges the code to produce a more efficient object program. A few compilers written for small computers will perform some optimization on the source program. It is doubtful if any compilers perform optimization at the object program level.

An example of program optimization which the alert programmer can utilize when non-optimizing compilers are used, is one of removing computations from inside a loop to the outside of a loop when the value does not change inside the loop.

For example, the following FORTRAN procedure contains values inside the loop which do not change:

```
DO 10 I = 1,100,1
TF = I
DB(I) = C1*C2*C3*TF/.7

10 CONTINUE
```

Relocating the constants external to the loop results in the following:

```
C = C1*C2*C3*10
DO 10 I = 1,100,1
TF = I
DB(I) = C*TF1.7

10 CONTINUE
```

A 36 percent reduction in execution time resulted for the second case when compiled under the non-optimizing compilers of the HEL CCS.

Another chance to optimize a loop procedure is one which has the following form:

```
DO I = 4, 100, 2
A1 = I*K
DB(I) = A1 + J(I)
```

This can be optimized, when K is invariant, and A is not altered elsewhere in the loop. The statement A = I*K can be modified to a procedure using addition (strength reduction), which executes faster than multiplication. The loop can be implemented in the following form:

```
A1 = 4*K
A2 = 2*K
DO I = 4, 100, 2
DB(I) = A1 + J(I)
A1 = A1 + A2
```

This optimization procedure works with integer values only. Loop routines which may permit REAL values would permit errors, due to roundoff, to accumulate as the loop is iterated. The reduction in execution time utilizing the strength reduction method is 4 percent on the HEL CCS system, a rather discouraging result when one is expecting a 50 percent improvement based upon integer multiply and addition execution time. DO loop overhead tends to swamp the expected improvement in this case.

Program Example

A good example of the application of some of the computer processing compromises is a dynamic flight display which HEL has adopted from a program developed for the Office of Naval Research (4).

The program is currently operating at an average 10-frame-per-second display rate with a picture complexity of 60 lines. This rate does not provide a sampling of the analog inputs from the HOT at 100 ms intervals; instead, analog data is sampled every 500 ms and the display data files are updated at the 10-frame-per-second rate by a frame synthesis (interpolation) procedure applied to display points in consecutive frames. A block diagram of the procedure is shown in Figure 6. The data base representing the 60 lines to be displayed is split into three blocks. The line endpoints are represented by an array of four-by-one vectors. Multiplication of the four-by-one vectors by the transformation matrix is carried out on each data block and distributed between the frame synthesis procedures.

The developers of this display technique were successful in providing smooth dynamic displays with an unsophisticated minicomputer system.

HEL has further analyzed this program development to determine what steps could be taken to increase the frame rate. Timing measurements were made on four procedures and it was determined that the transform matrix multiplications require 40 percent of the processing time (Appendix F).

A fast array processor would reduce the transform matrix multiplication time sufficiently to permit a 2:1 improvement in the frame rate of the example program. However, equipment presently available at HEL could provide the same improvement by using parallel processing in which the various routines would be divided between two computers. Other steps which can be implemented would be the use of a more efficient FORTRAN compiler than is presently available and the addition of a hardware floating-point arithmetic unit.

Inputs to the visual system for derivation of the visual cues come from various pick-off points in the simulator computer. The real-time response of the visual system presents two problems. The first requires a determination of the allowable or optimum visual delay relative to the motion system response. It can neither lead nor lag the motion system by too much before it disturbs the pilot. The second problem is one of maintaining the minimum processing time around the visual system loop represented by blocks 1, 2, 4a, 5a, 6, 7 and 8 in Figure 5. As the visual system loop time approaches 100 ms, there will be very little time to play with in optimizing the correlation of the visual output to the physical motion.

To improve the frame rate or to allow an increase in picture complexity at the existing frame rate, it will be necessary to process data in parallel as indicated by blocks 4a, 4b, 5a, 5b, in Figure 5. An alternative to this approach would be a special high-speed arithmetic processor.

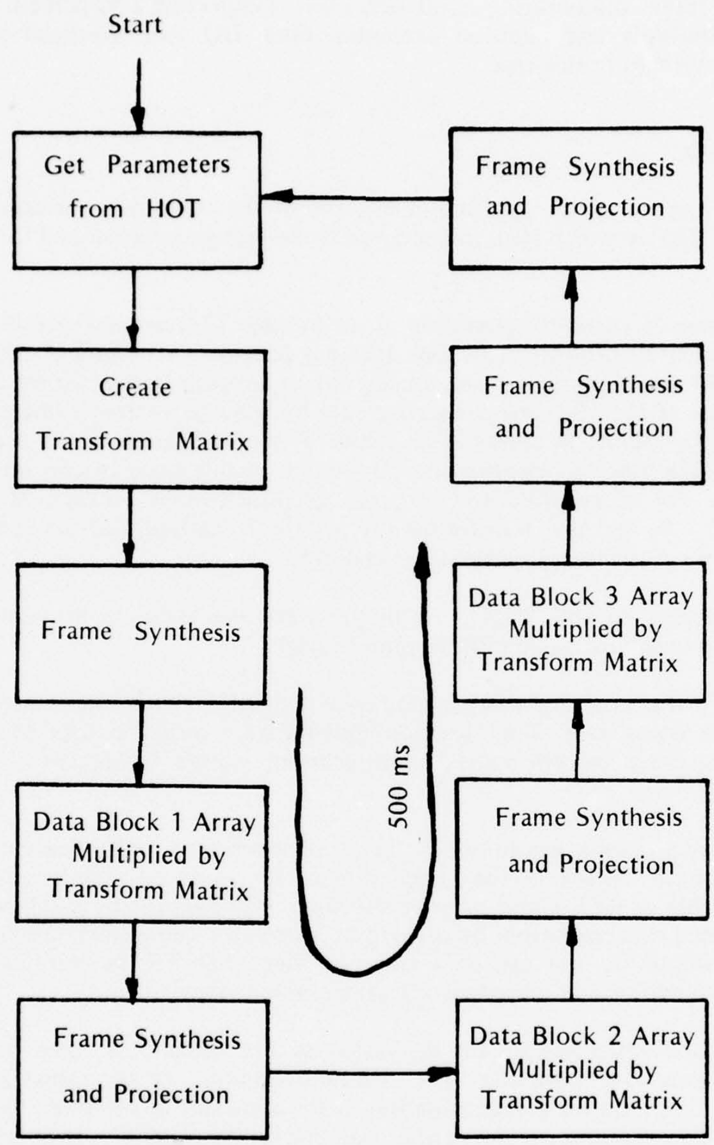

Figure 6. Dynamic flight display processing loop.

Real-time systems having multilevel interrupt capability can provide an opportunity to reduce the quantity of code and improve execution speed if multiple registers are provided at each interrupt level thus eliminating the need for the software to transfer processing unit registers to memory when servicing an interrupt. The IBM system/7 is a good example of multiple level interrupt hardware.

To summarize those points critical to getting an imaging application up to speed, look at the efficiency of the compiler with respect to arithmetic processing speed, and, if the opportunity is available, select the computer that has the greatest number of accessible general-purpose registers, larger word size, hardware floating-point arithmetic units, multiple interrupt registers, and microcode capability. This does not require selection of a system having IBM 360/370 level of capability. Systems Engineering Laboratories SEL 32/55, Interdata 8/32, Harris Slash 6 and Slash 7 are some good examples of low cost systems with exceptional capability.

HARDWARE INTEGRATION PROBLEMS

An unfortunate situation exists at HEL regarding the location of the CCS and the HOT; they are separated by a distance of 250 feet. Also, the HOT was not specifically designed for the type of imaging system which HEL has interfaced to it. A number of electrical/electronic problems required solution before successful displays could be generated. These problems were related to AC power systems, analog output signals, data transmission over 250 feet of cable, and real-time response problems. The solutions to these problems are considered next.

In addition to the complications provided by the 250 foot physical separation between the HOT and CCS, there exists an AC power system problem which results from each system being powered from different AC transformers. Severe ground loop signals develop from this situation. Figure 7 characterizes the HEL system situation. Common mode noise sources, shown as an e_{cm} signal, are due to potential differences between the two ground systems. Other common mode signals may develop in the cable due to proximity to high current carrying lines along the cable run.

Noise was also observed on the analog outputs from the HOT in addition to those cited above. These noise signals consisted of low frequency periodic noise, high frequency sinusoidal bursts and random transient spikes.

Delicate circuits of both systems are protected and the data precision retained by, first, isolating the two systems, second, filtering the HOT noise, and third, conditioning the HOT analog outputs to eliminate undesirable signal types and standardize them where possible. The third precedure can significantly reduce the amount of data processing required to quantify a parameter.

The first step in eliminating noise sources along the cable route was to select a suitable cable type. Twisted pair performs reasonably well in these situation; however, when in the vicinity of other cables which may radiate high energy impulse noise, there is always some pick-up. Belden type 8773 cable was selected for providing the desired twisted pair and also for providing a shield for each twisted pair. Shields were terminated at the HOT system ground.

The second step in eliminating the noise components required the use of a differential line receiver and dual-line drivers. Dual differential 747 operational amplifiers were used for both line divers and receivers. The method, shown in Figure 8, also provided isolation between the two systems.

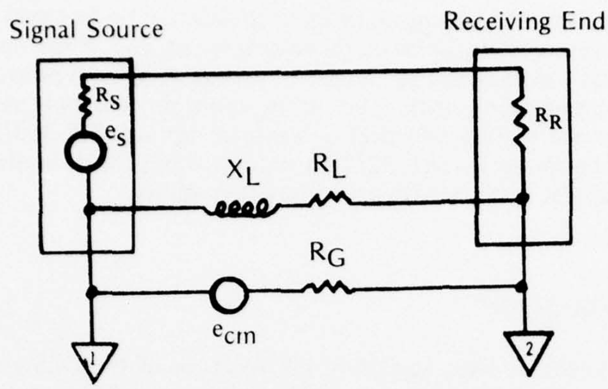

Figure 7. System signal equivalent.

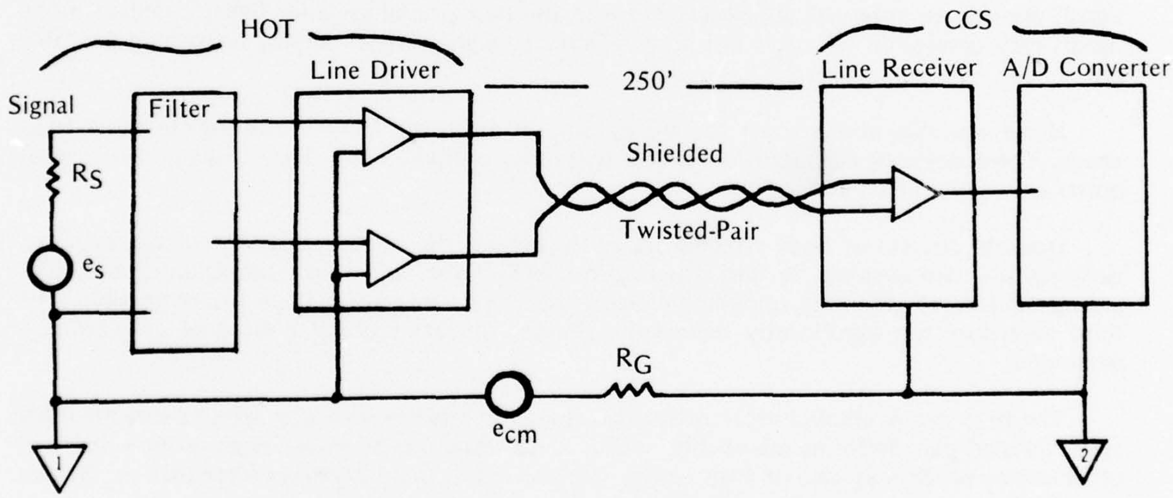

Figure 8. Differential line driver/receiver configuration.

Selected RC filters and integrators were utilized to eliminate HOT noise and to provide DC levels corresponding to rate of climb, altitude, airspeed, etc.

Appendix E provides some representative data analysis of the HOT analog output after digitization at the CCS end. Data samples were obtained with the HOT electronics and hydraulic system on and the simulator "on the ground."

High-speed video data and CRT beam positioning commands sent from the CCS to the HOT are carried via RG180 coaxial cable placed parallel to the analog cable. The display unit signal electronics are isolated from the HOT system ground and referenced to the CCS system ground.

Slight character and vector degradation was noticable in the form of reduced intensity at the beginning of each line causing gaps to appear in what is normally a smoothly stroked vector or character. The gaps resulted from line capacitance and high input impedence at the CRT video amplifier which allowed the video signal rise time to become excessive. Reducing the video amplifier input impedence to 100 ohms was sufficient to improve the video rise time and restore the integrity of the vectors and characters.

The original CCS displays were located at the end of 30- and 100-foot cable lengths. Locating a display in the HOT at a distance of 250 feet required readjustment of the display system for best display presentation at this HOT end. This has resulted in vector endpoint closure problems for the displays located at the end of shorter cable lengths. Delay lines will be provided for the X, Y, and Z inputs to the original CCS displays to compensate for differing cable lengths.

CONCLUSIONS

Much has been learned in integrating the helicopter simulator to provide real-time dynamic, computer-generated visual displays. The photographs that are included in this report demonstrate the capability of the installation.

Figure 1 shows the Dynamic Flight Display presentation as seen in the HOT. A close-up of the CRT presentation is shown in Figure 9.

Figures 10 and 11 show two early versions of integrated avionics CRT displays. Several other versions are currently being tested.

Figure 12 is a plasma panel display of the HOT ground track during a test mission.

Figures 13 and 14 are photographs of simulated NOE images which are to be presented on the panel mounted raster-scan system.

Development and maturing of this capability will provide HEL with a method of evaluating through simulation, many types of panel-mounted displays. Helmet-mounted displays will be added in the near future.

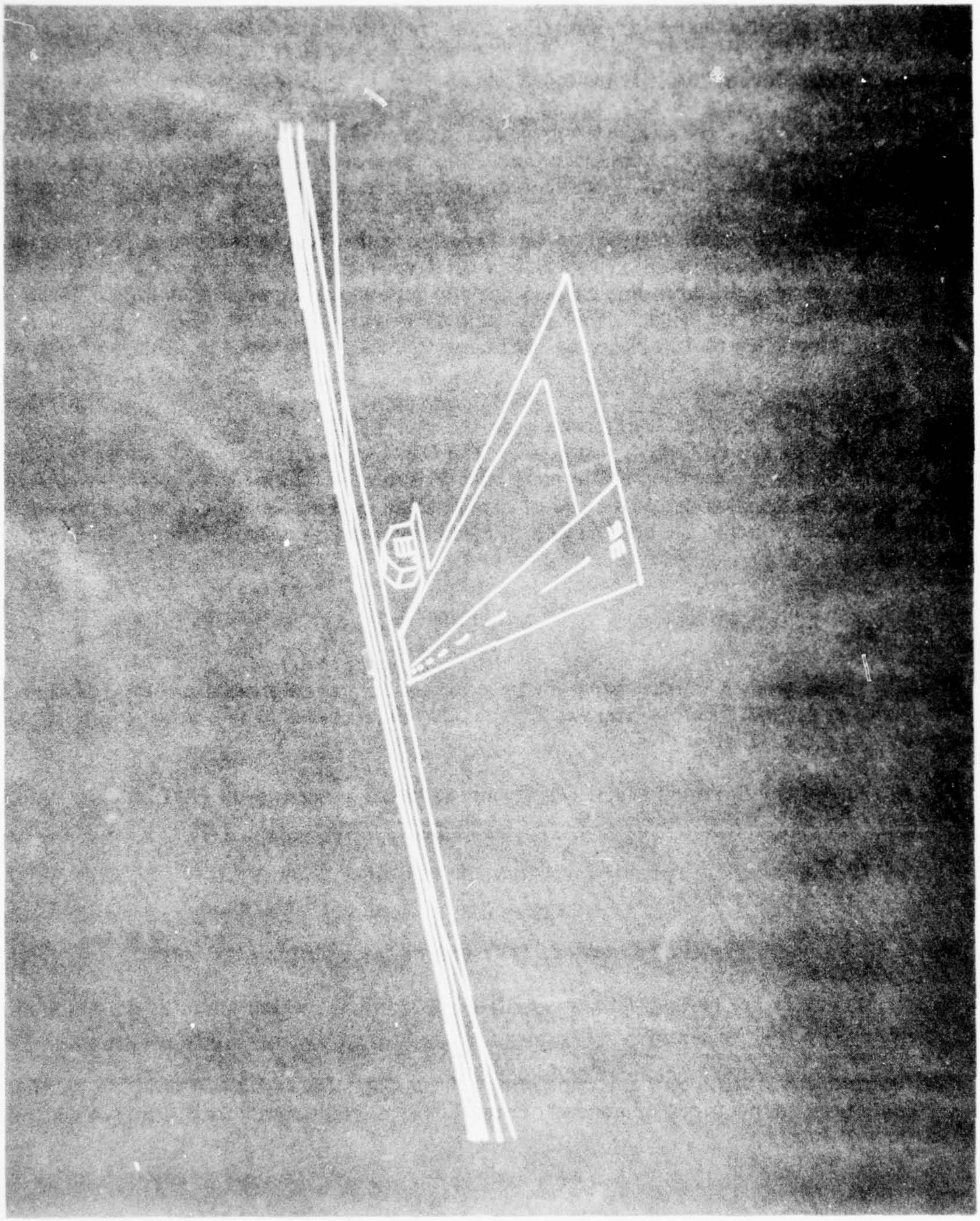

Figure 9. Closeup of the dynamic flight display presentation.

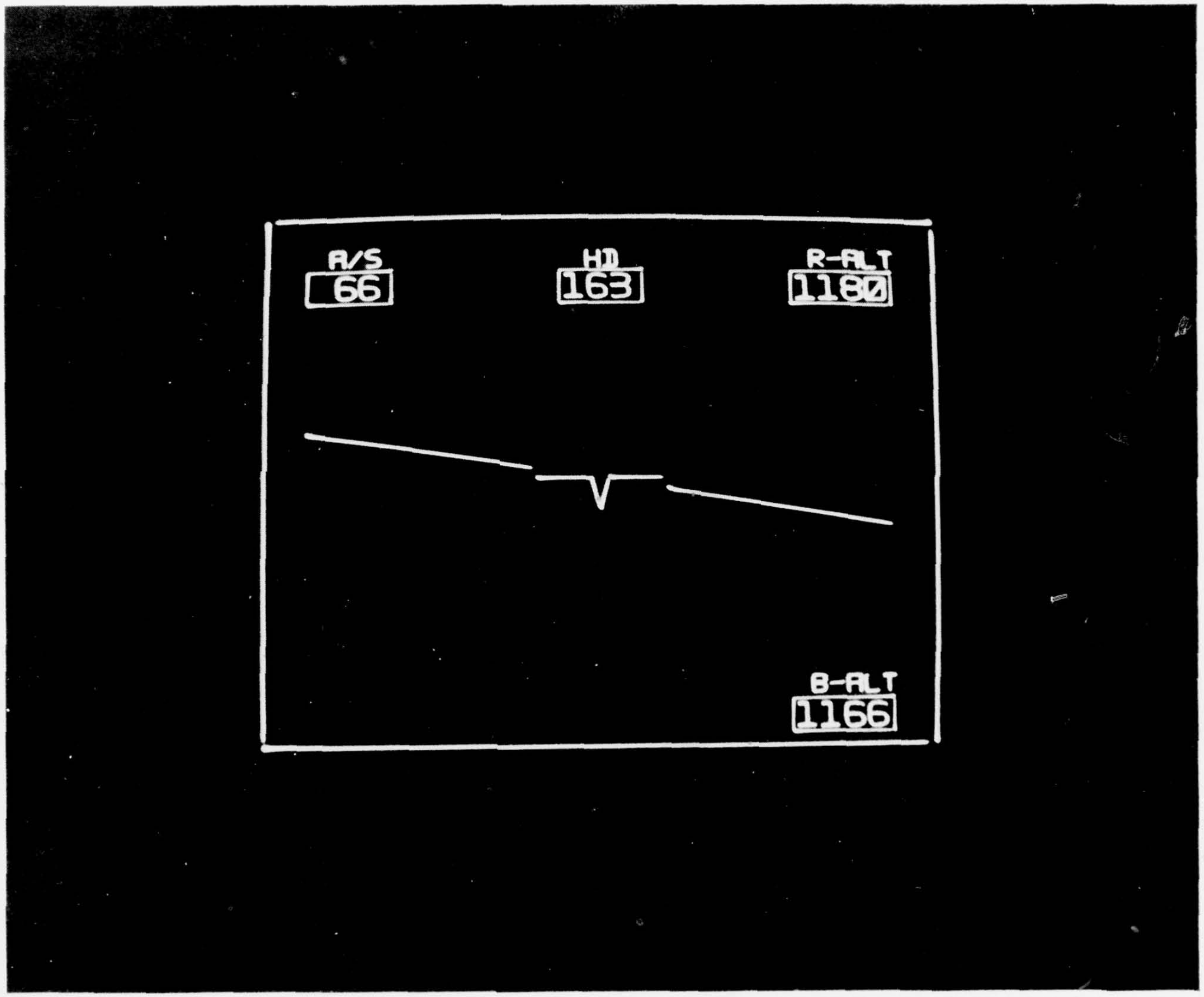

Figure 10. Integrated avionics CRT display, version I.

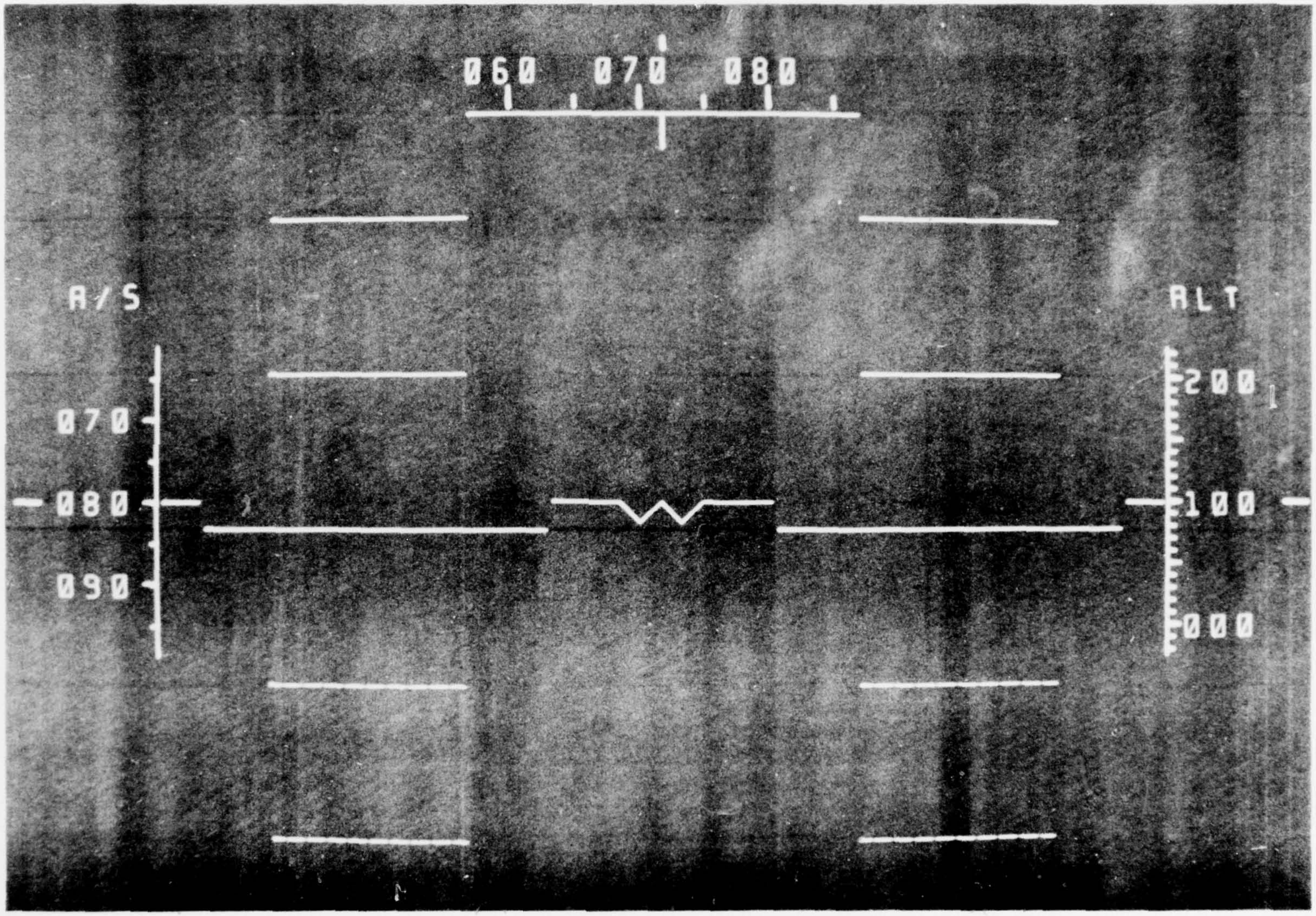

Figure 11. Integrated avionics CRT display, version II.

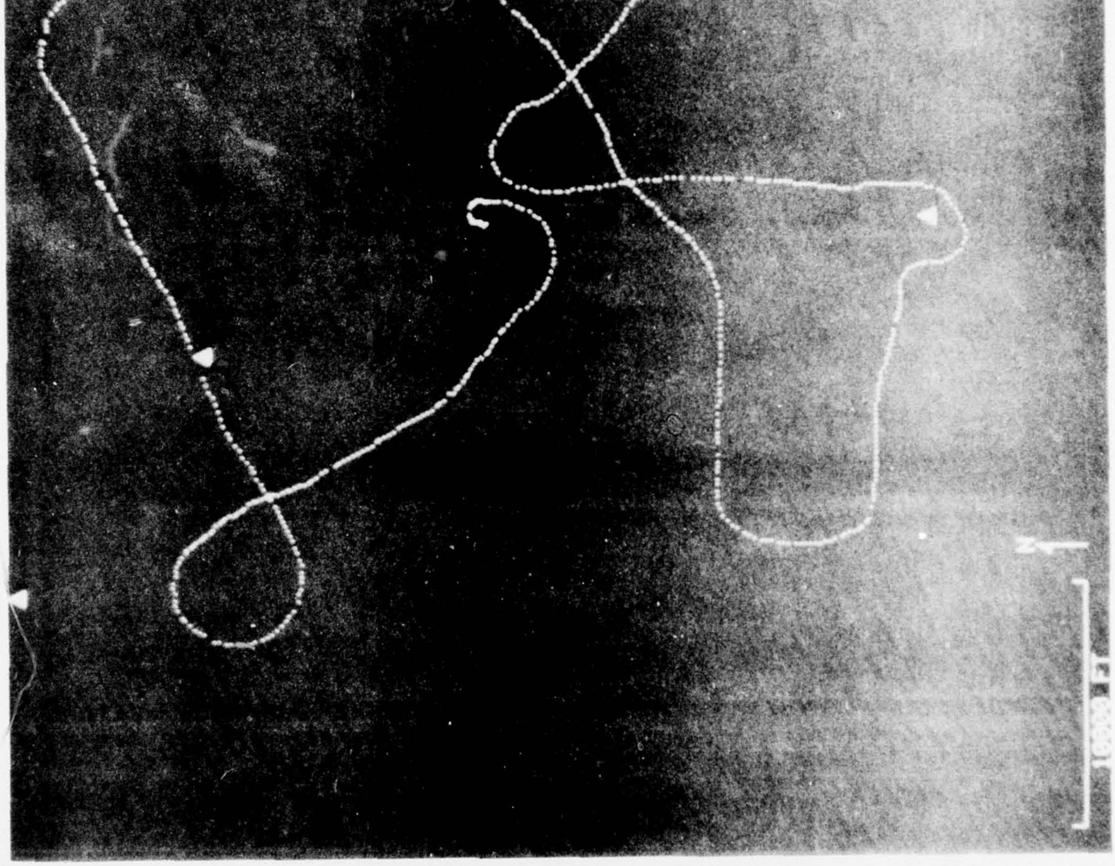

Figure 12. HOT ground track presented on plasma panel.

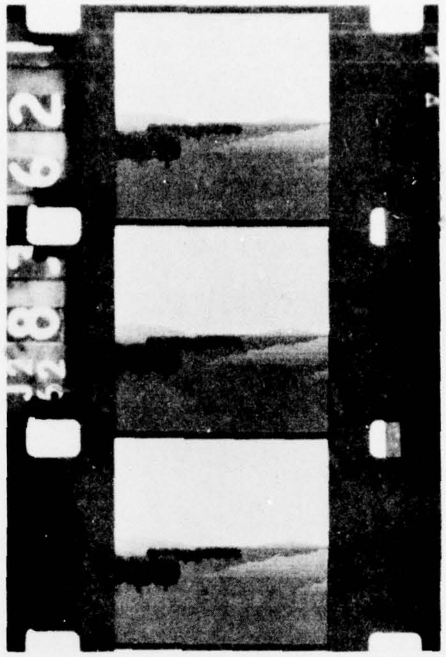

Figure 13. Simulated NOE raster scan display.

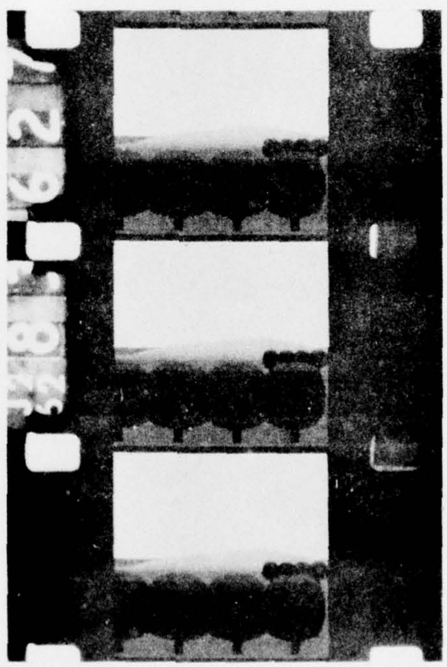

Figure 14. Simulated NOE raster scan display.

REFERENCES

1. Arden, B.W. An introduction to digital computing. Addison-Wesley, Reading, MA, 1963.

2. Artwick, B. A versitile computer-generated dynamic flight display. Aviation Research Laboratory, ARL-76-5/ONR-76-1, University of Illinois, Savoy, IL, May 1976.

3. Herald, G.L. Digivue gas display controller for Varian 620 computers. Technical Note 5-75, U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, MD, March 1975.

4. Herald, G.L. Interactive graphics data preview. Technical Note 10-75, U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, MD, Dec 1975.

5. Singer-Link Division. Description of HOT helicopter operational trainer. Report No. 7304-102, Binghamton, NY, Mar 1971.

6. Human Engineering Applications Directorate, Team 4. Interactive graphics—a tool for applied human factors engineering. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, MD, Jun 1977.

7. Naval Training Equipment Center. 8th NTEC/industry conference proceedings, Orlando, FL, 1975.

APPENDIX A

HELICOPTER OPERATIONAL TRAINER CHARACTERISTICS

Basic HOT Instrument Characteristics

Airspeed	0 to 140 kts
Altimeter	-2000 to 18,000 ft
Rate of Climb	0 to ± 2500 ft/min
Turn Indicator	2 minute turn
Percent Torque	0 to 110 percent
Engine Tachometer	0 to 3500 rpm
Rotor Tachometer	0 to 350 rpm

Motion System Kinesthetic Cues[a]

	Limit	Acceleration	Velocity
Pitch	$\pm 11^{\circ}$	$\pm 25^{\circ}/sec^2$	$+9.16^{\circ}/sec$ $-14.35^{\circ}/sec$
Roll	$\pm 11^{\circ}$	$\pm 50^{\circ}/sec^2$	$\pm 25^{\circ}/sec$

[a]Kinesthetic cues are synchronized with instrument indications

Other Cues

Rough Air
Landing Impact
Rotor Induced Vibration

Controls

Flight: Cyclic Stick, Collective Stick, Directional Pedal

Powerplant: Engine on/off switch, Throttle

APPENDIX B

IDIIOM CALLIGRAPHIC DISPLAY SPECIFICATIONS

CRT (monochrome)	Phosphor -P31
Deflection	Magnetic
Resolution	.015" Spot Size
Character Writing Time	10 us
Vector Writing Time	50 us full screen
Circle Writing Time	100 us
Position Generator Resolution	1024x by 1024Y
Functions	Blink control Character Rotate 90^{o} CCW 4 levels intensity Line structure-dot, dash, dot-dash, solid
Brightness	40 FL at 100k in/sec writing speed, 60Hz frame rate

APPENDIX C

RASTER SCAN IMAGING SYSTEM SPECIFICATIONS

1. 512 x 640 horizontal raster-scan 2.1 interlace
2. 16 levels grey scale, 1 level alphanumeric overlay
3. 4096 colors
4. Image manipulation functions
    a. windowing
    b. translation
    c. scaling
    d. zooming
    e. scrolling
    f. reversal rotation
5. Graphic functions
    a. vectors
    b. conics
    c. plots
6. Characters - 7 x 9 font

APPENDIX D

DIGIVUE SPECIFICATIONS

Individually Addressable Light Points	262,144
Character Capacity	
with 5x7 matrix	4,335
with 7x9 matrix	2,223
Dot Spacing	.0167" center-to-center
Light Spot Size	7.5–8.5 mil
Vector Address Rate	50K/dots per second
Viewing Angle	160°
Brightness	50 ft./L approximately
Contrast Ratio-Small Area	25:1 nominal
Light Spectrum	Neon orange (5852A° predominant)
Bulk Erase	20 microseconds
Operating Temperature Range	0°C to +55°C
Storage Temperature Range	-62°C to +85°C
Addressing Rate	
Serial	1,400 characters/sec 5x7
Parallel	10,000 characters/sec 5x7
Logic Level	TTL
Clock	Synchronous or asynchronous
Overall Unit Size	16.5" x 15.5" with hollow rear proj port
Panel Size	12.25" x 12.25"
Active Display Area	8.55" x 8.55"
Character Size	
5x7	80 x 120 mils
7x9	120 x 150 mils
Power Supply Input Requirements	1.8 amps maximum

APPENDIX E

SIGNAL ANALYSIS OF HOT ANALOG SIGNALS

HOT Electronic and Hydraulic System on Simulator on "Ground"

Signal	Chan.	Min. (mv)	Avc. (mv)	Max. (mv)	Spred (mv)
Lat. vel. gnd	1	-	-	-	-[a]
Airspeed	2	-49	-34	-27	22
COS Heading	3	9009	9016	9026	17
Altitude	4	-37	-32	-29	8
Pitch	5	-1123	-1118	-1116	7
Bank	6	-7	7	15	8
Sin Heading	7	-2244	-2224	-2212	32
Turn Rate	8	-	-	-	-[a]
Cyclic Pos. Lat.	9	-5537	-5522	-5508	29
Cyclic Pos. Long.	10	-7515	-7488	-7476	39
Collective	11	671	686	701	30
Pedal	12	-3813	-3792	-3779	34
Rotor Thrust	13	752	789	820	68
Long. Vel. Gnd.	14	-17	-2	20	37
Slip	15	10	24	39	29
Vert. Vel	16	-2	5	10	12

[a]Defective at time of measurement.

APPENDIX F

PROCESS TIME FOR THE DYNAMIC FLIGHT DISPLAY PROGRAM TIMED ON THE VARIAN 620f-100 USING THE INTERNAL TIME CLOCK

Array Processing Procedure (28 points)	51 ms
Frame Synthesis Procedure (120 points)	6 ms
Transform Matrix Development	5 ms
A/D Conversion and Parameter Calculations (6 values)[a]	13 ms
Display Loop Timing	
A/D Conversion and Parameter Calculations (1 call)	13 ms
Frame Synthesis (5 calls)	30 ms
Transform Matrix (1 call)	5 ms
Array Processing (3 calls - 120 points)	213 ms
All other procedures	239 ms

[a]Heading (2 values), Bank, Pitch, Airspeed, Altitude

APPENDIX G

SUBROUTINES USED TO DETERMINE ARITHMETIC EXECUTION TIME

Integer Routine Used to Compare Standalone and Operating System Arithmetic Execution Speed

```
/FORT
      SUBROUTINE FXPT
      DATA IZ1/7500/
      DATA IZ2/7900/
      DATA IX1/6500/
      DATA IX2/7000/
      DATA IY1/10/
      DATA IY2/100/
      DO 10 I=1,32000
      IT=(IZ1-IX1)/((IX2-IX1)-(IZ2-IZ1))
      IZ1=IT*(IZ2-IZ1)+IZ1
      IX1=IZ1
      IY1=IT*(IY2-IY1)+IY1
      IA=IY1
      IB=IX1
      IC=IT
10    CONTINUE
      END
```

Floating Point Routine Used to Compare Standalone and Operating System Arithmetic Execution Speed

```
      SUBROUTINE PUSH
      DATA X1/6500./
      DATA X2/7000./
      DATA Y1/10./
      DATA Y2/100./
      DATA Z1/7500./
      DATA Z2/7900./
      DO 10 I=1,32000
      T=(Z1-X1)/((X2-X1)-(Z2-Z1))
      Z1=T*(Z2-Z1)+Z1
      X1=Z1
      Y1=T*(Y2-Y1)+Y1
      A=Y1
      B=X1
      C=T
10    CONTINUE
      END
```

Mixed Integer/Floating Point Routine Used to Compare Standalone And Operating System Arithmetic Execution Speed (No Square Root or Absolute Value)

```
      SUBROUTINE MX1
      DATA X1/6500./
      DATA X2/7000./
      DATA Y1/10./
      DATA Y2/100./
      DATA Z1/7500./
      DATA Z2/7900./
      DATA IZ1/7500/
      DATA IZ2/7900/
      DATA IX1/6500/
      DATA IX2/7000/
      DATA IY1/10/
      DATA IY2/100/
      DO 10 I=1,32000
      IT=(IZ1-IX1)/((IX2-IX1)-(IZ2-IZ1))
      IZ1=IT*(IZ2-IZ1)+IZ1
      IX1=IZ1
      IY1=IT*(IY2-IY1)+IY1
      IA=IY1
      IB=IX1
      IC=IT
10    CONTINUE
      DO 20 J=1,32000
      T=(Z1-X1)/((X2-X1)-(Z2-Z1))
      ZI=T*(Z2-Z1)+Z1
      XI=Z1
      YI=T*(Y2-Y1)+Y1
      A=Y1
      B=X1
      C=T
20    CONTINUE
      END
```

Mixed Integer/Floating Point Routine Used to Compare Standalone
And Operating System Arithmetic Execution Speed
(Includes Square Root and Absolute Value)

```
      SUBROUTINE  MX2
      DATA  X1/6500./
      DATA  X2/7000./
      DATA  Y1/10./
      DATA  Y2/100./
      DATA  Z1/7500./
      DATA  Z2/7900./
      DATA  IZ1/7500/
      DATA  IZ2/7900/
      DATA  IX1/6500/
      DATA  IX2/7000/
      DATA  IY1/10/
      DATA  IY2/100/
      DO 10  I=1,32000
      IT=(IZ1-IX1)/((IX2-IX1)-(IZ2-IZ1))
      T=(Z1-X1)/((X2-X1)-(Z2-Z1))
      A=ABS(T)
      IA=IABS(IT)
      B=SQRT(T)
10    CONTINUE
      END
```